Automatic Referent Resolution of Deictic and Anaphoric Expressions

Carla Huls*
University of Nijmegen

Edwin Bos*
University of Nijmegen

Wim Claassen*
University of Nijmegen

Deictic and anaphoric expressions frequently cause problems for natural language analysis. In this paper we present a single model that accounts for referent resolution of deictic and anaphoric expressions in a research prototype of a multimodal user interface called EDWARD. The linguistic expressions are keyed in by a user and are possibly accompanied by pointing gestures. The proposed model for reference resolution elaborates on Alshawi's (1987) notions of context factors and salience and integrates both linguistic and perceptual context effects. The model is contrasted with two alternative referent resolution models, namely, a simplistic one and the more sophisticated model proposed by Grosz and Sidner (1986). Based on empirical and analytical grounds, we conclude that the model we propose is preferable from a computational and engineering point of view.

1. Introduction

This paper deals with the automatic referent resolution of deictic and anaphoric expressions in a research prototype of a multimodal user interface called EDWARD. The primary aim of our project is the development and the assessment of an interface that combines the positive features of the language mode and the action mode of interaction (Claassen et al. 1990). EDWARD (Huls and Bos 1993; Bos et al. 1994) integrates a graphical graph-editor called Gr^2 (Bos in press) and a Dutch natural language (NL) dialogue system called DoNaLD (Claassen and Huls 1991). One of the application domains involves a file system environment with documents, authors, a garbage container, and so on. The user can interact with EDWARD by manipulating the graphical representation of the file system (a directed graph), by menus, by written natural or formal language, or by combinations of these. EDWARD responds in NL (either written or spoken) and graphics.

In this paper we will go into the semantic and pragmatic processes involved in the referent resolution of deictic and deixis-related expressions by EDWARD. (Syntactic issues will not be discussed here; for these, see Claassen and Huls 1991.) The proper interpretation of deictic expressions depends on the identity of the speaker(s) and the audience, the time of speech, the spatial location of speaker and audience at the time of speech, and non-linguistic communicative acts like facial expressions and eye, hand, and body movements. Lyons (1977, p. 637), provides the following definition of deixis:

* Nijmegen Institute for Cognition and Information, P.O. Box 9104, 6500 HE Nijmegen, The Netherlands. E-mail: huls@nici.kun.nl

> the location and identification of persons, objects, events, processes and activities being talked about, or referred to, in relation to the spatiotemporal context created and sustained by the act of utterance and the participation in it, typically, of a single speaker and at least one addressee.

In the context of the present paper, we distinguish three types of deixis: personal, temporal, and spatial deixis. Personal deixis involves first- and second-person pronouns (e.g., *I, we,* and *you*). Temporal deixis is realized by the tense system of a language (e.g., *he lives in Amsterdam*) and by temporal modifiers (e.g., *in an hour*). Temporal deixis relates the time of speech to the relation(s) expressed by the utterance. Spatial deixis involves demonstratives or other referring expressions that are produced in combination with a pointing gesture (e.g., *this*↗ *file,* in which ↗ represents the pointing gesture). In the present paper, most attention will be given to spatial deixis.

Deictic expressions can be contrasted with anaphors. Unlike deictic expressions, anaphors can be interpreted without regard to the spatiotemporal context of the speaking situation. Their interpretation depends merely on the linguistic expressions that precede them in the discourse. For example, *this* is an anaphor in *Print the file about dialogue systems. Delete this.* In many languages, the words used in deictic expressions are also used in anaphoric expressions.

Deictic and anaphoric expressions frequently cause problems for NL analysis. Sijtsma and Zweekhorst (1993) find referent resolution errors in all three commercial NL interfaces they evaluate. In research laboratories, a couple of systems capable of interpreting deictic expressions recently have been developed. Allgayer et al. (1989) describe XTRA, a German NL interface to expert systems, currently applied to supporting the user's filling out a tax form. XTRA uses a dialogue memory and a tax-form hierarchy to interpret multimodal referring expressions. Data from the dialogue memory and from gesture analysis are combined (e.g., by taking the intersection of two sets of potential referents suggested by these information sources). Neal and Shapiro (1991) describe a research prototype called CUBRICON, which combines NL (English) with graphics. The application domain is military tactical air control. Like XTRA, CUBRICON uses two models to interpret deictic expressions: an attentional discourse focus space representation (adapted from Grosz and Sidner 1986) and a display model. Stock (1991) describes ALFresco, a prototype built for the exploration of frescoes, using NL (Italian) and pictures. For referent resolution in ALFresco, topic spaces (Grosz, 1978) are combined with Hajičová's (1987) approach, in which entities are assumed to "fade away" slowly. Cohen (1992) presents Shoptalk, a prototype information and decision-support system for semiconductor and printed-circuit board manufacturing with a NL (English) component. In Shoptalk too, the interpretation process is based on the approach of Grosz and Sidner. We believe that the fact that these systems use two separate mechanisms for modeling linguistic and perceptual context is a disadvantage over the use of only one mechanism for referent resolution. From a computational and an engineering position, one mechanism that handles both deictic and anaphoric expressions in the same way is preferable.

We will (try to) show how both deictic and anaphoric references can be resolved using a single model. We have used the framework presented by Alshawi (1987) to develop a general context model that is able to represent linguistic as well as non-linguistic effects on the dialogue context. This model is used, in conjunction with a knowledge base, by EDWARD's interpretation component to solve deictic and anaphoric referring expressions. The same model and knowledge base are used by EDWARD's generation component to decide the form (e.g., *he, the* writer, *a* man), the

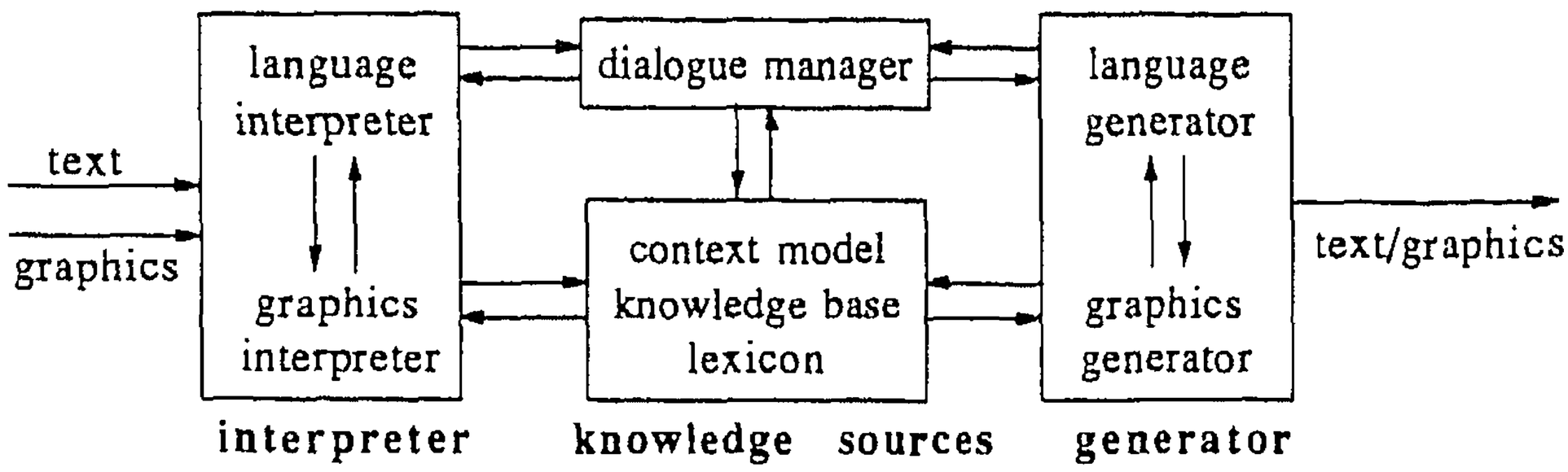

Figure 1
The main components of EDWARD.

content (e.g., the *writer*, the *husband*), and the mode (e.g., linguistic or simulated pointing gesture; Claassen 1992; Claassen et al. 1993) of referring expressions. In this paper, however, we focus on the use of the context model to resolve deictic and anaphoric expressions keyed in by the user.

The rest of this paper is structured as follows: in Section 2, we present an overview of EDWARD. Next, we describe the knowledge sources EDWARD uses to interpret deictic and anaphoric expressions (Section 3). In Section 4, we go into the process of interpreting deictic and anaphoric expressions in some detail. Subsequently, in Section 5, we present some user interactions with EDWARD and we compare the results of EDWARD's referent resolution model with two other models including that of Grosz and Sidner (1986).

2. Overview of EDWARD

EDWARD is implemented in Allegro Common Lisp and runs on DECstations. Figure 1 presents a schematic overview of EDWARD's system architecture. The arrows represent the information flow between the main components. EDWARD accepts input from two devices: keyboard and mouse device. The output is directed to two devices on the screen: a NL output text window and a graphics display, and, optionally, to a speech synthesizer. The dialogue manager coordinates input and output expressions and controls the linguistic and graphics processes. It maintains the Context Model, the knowledge base, and the lexicon; in addition, it decides which individual instances stored in the knowledge base must be represented on the graphics display, and it makes sure that the display is always up to date. The language interpreter and the language generator consult the Context Model, the knowledge base, and the lexicon. Both the interpreter and the generator operate in an incremental fashion. Figure 2 illustrates how the user can interact with EDWARD.

The area occupying most of the screen is the graphics display: a window called Modelwereld (Model World). The tree shown in Figure 2 represents a hierarchy of directories (depicted as bookcases) and files (e.g., reports, papers, e-mail messages, and books). The viewport shows only part of the Model World window, which in principle extends indefinitely. In the bottom–left corner of the viewport, a garbage container and a copier are displayed. The bear icon, at the bottom in the middle, represents the system itself (i.e., EDWARD). Using a mouse, the user can manipulate the graphical representation of the domain objects by pointing, clicking, and dragging. At the bottom of the Model World window, a mouse documentation bar is presented (the

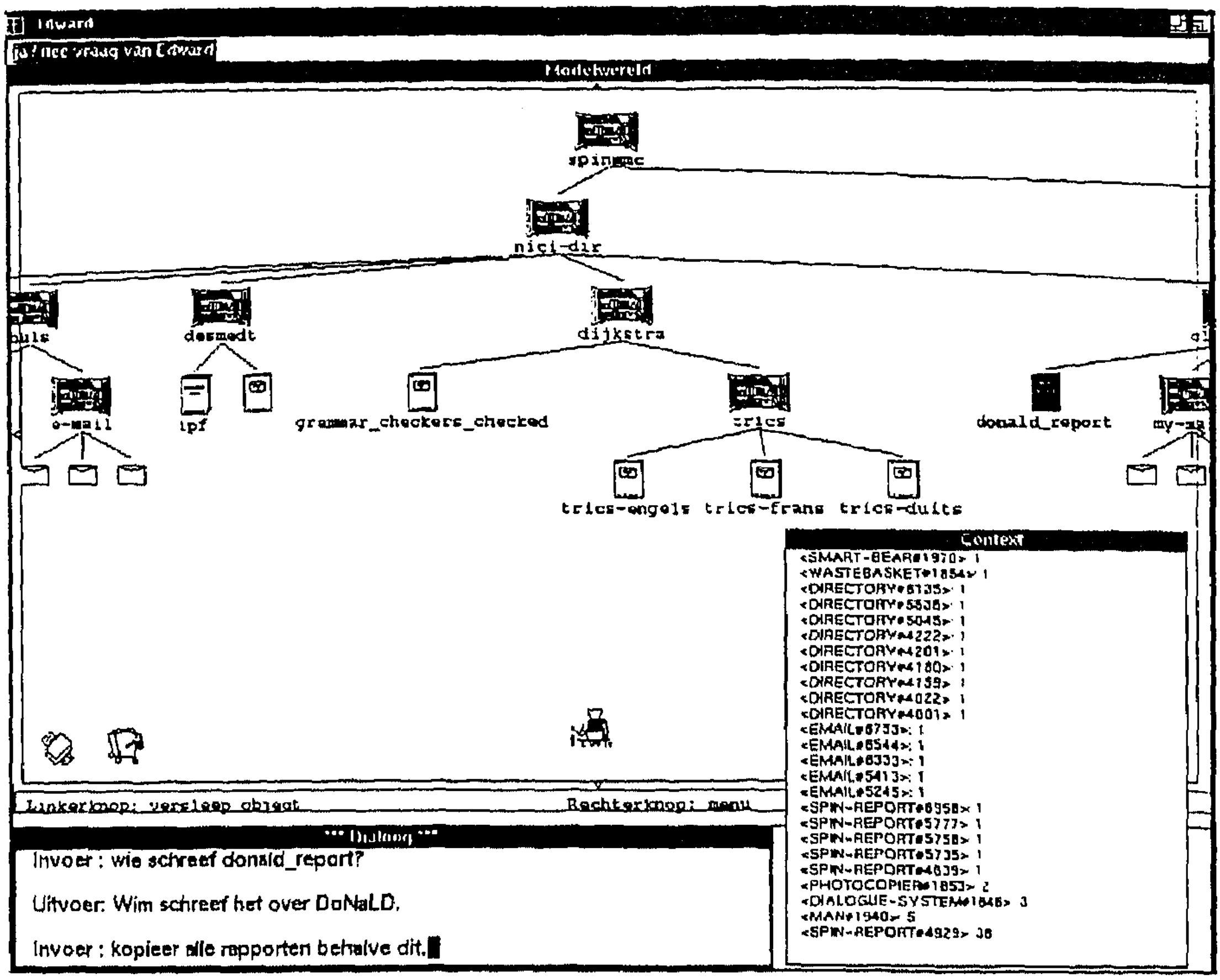

Figure 2
A screen dump of EDWARD. The user is entering the command: *Kopieer alle rapporten behalve dit.* (Copy all reports except for this one.) after selection of the file icon labeled donald_report.

Dutch word *Linkerknop* means 'left button,' *versleep* means 'to drag,' and *Rechterknop* means 'right button'). In the bottom–left area of the screen is the NL interaction window labeled Dialoog (Dialogue). Here the user can enter NL commands, questions, or assertions. Depending on the number of words and ambiguities in a linguistic expression, interpretation takes between 0.5 and 1.5 seconds when running on a Personal DECstation 5000. In Figure 2, the user has requested the system to *copy all reports except for this one.* At the bottom right, the trace window Context displays the salience values of some of the discourse referents. Referents are presented by the name of the concept class they belong to, followed by the number sign (#) and a unique number enclosed in angle brackets, e.g., <directory#4001> and <spin-report#4929> (spin-reports are a special kind of project reports).

3. Knowledge Sources

To be able to interpret referring expressions, EDWARD uses three knowledge sources: a knowledge base, a context model, and a lexicon. The knowledge base stores the permanent generic and specific world knowledge of the system, whereas the Context Model temporarily "memorizes" which individual instances from the knowledge base have been referred to in the dialogue. The lexicon specifies morphophonological and syntactic features of words and contains links between words and the knowledge base

that represents lexical meaning. In this section we will describe the knowledge base and Context Model.

3.1 The Knowledge Base

The knowledge base is a semantic network implemented in CommonORBIT (De Smedt 1987), a frame- based language somewhat similar to KL-ONE (Brachman and Schmolze 1985). The nodes in the network represent classes and instances of entities and relations. For example, the class <*person*> contains two subordinate classes, <*man*> and <*woman*>, and the concept of sending an object to someone is represented by a generic relation called <*send*>. Individual objects in the domain are represented by instances; e.g., an individual who is a man might be represented as <man#24>. If he sends a message, a relation instance is created; e.g., <send#89>. Contrary to KL-ONE, relations have a time interval associated with them, which represents the period of time during which the relation is assumed to hold. A time interval has a start value and an end value. The end value may be *NOW*, which is a dynamic value representing open-endedness in a time interval. Much like in KL-ONE, relations[1] contain role-filler class restrictions and role-set restrictions. For example, with the generic relation <*send*>, three semantic (case) roles are associated, called <agent>, <goal>, and <recipient>. The role-filler class restrictions then specify, for example, that the fillers of the <agent> and <recipient> roles must be either persons or institutions and that the filler of the <goal> role (the object that is sent) must be concrete and excludes persons. This information is used by the interpretation component to restrict the referent sets of the role fillers of a relation. The role-set restrictions specify, for example, that the filler of the <recipient> role in a <*send*> relation is not, at least not in our current domain, allowed to be identical to the filler of the <agent> role.[2] The interpreter could use these restrictions to exclude certain referents from the set of potential referents.

Depending on the domain EDWARD is being applied to, a filter is defined to determine which concepts of the knowledge base should be visually represented on the screen. The file system domain filter, for instance, allows instances of particular file system classes, such as directories, e-mail messages, reports, and books. The instances passing the filter are represented by icons that depict their class. The only relation instances passing the file system domain filter are <*contain*> relations and <*name*> relations. A <*contain*> relation is represented graphically by a straight line linking the icon that represents the container and the icon representing the object contained. <*Name*> relations (if present) are represented by a label underneath the icon of the named object (see Figure 2).

3.2 The Context Model

The second knowledge source EDWARD uses to analyze referring expressions is the Context Model. The central notion in this model is **salience**. The intuitive notion of salience has two important characteristics. In the first place, the salience of an instance at a given moment is determined by a diversity of factors of varying importance. In written language, recency of mention is known to be an important factor, as are syntactic and semantic parallelism, the markedness of expressions and constructions, and so on. Spoken language adds intonation, and when the situational context gets involved, various perceptual factors like visibility join in. The second important characteristic

---

1 KL-ONE represents relations by "roles" that correspond to two-placed predicates.

2 Usually, the notion of C-command is used to handle these kind of restrictions. However, this syntactic solution works only for restrictions *within one sentence*. The role-set restriction approach we propose is independent of the size of units processed by the interpreter.

Table 1
Context factors and their significance weights after successive updates.

Context Factors	Objects in Scope	Successive Weights
Linguistic CFs		
Major-constituent referents CF	Referents of subject, (in)direct object, and modifier	[3, 2, 1, 0]
Subject referent CF	Referents of the subject phrase	[2, 1, 0]
Nested-term referent CF	Referents of noun phrase modifiers (e.g., prepositional phrase, relative clause)	[1, 0]
Relation CF	Relations expressed by subject, prepositional phrase, and relative clause	[3, 2, 1, 0]
Perceptual CFs		
Visible referent CF	Referents visible in the current viewport	[1, . . . , 1, 0]
Selected referent CF	Referents selected in the model world	[2, . . . , 2, 0]
Indicated referent CF	Referents indicated by a pointing gesture	[30, 1, 0]

of salience is its gradedness. An individual instance may be more or less salient, may gradually become less salient, etc.

Alshawi (1987) provides a general framework for modeling salience that does justice to both characteristics mentioned above. The central construct in this framework is that of **context factor** (CF). A CF is defined by a scope, which is a collection of individual instances; a significance weight, represented by an integer; and a decay function, which indicates by what amount the CF's significance weight is to be decreased at the next update. In EDWARD we have adopted Alshawi's notion of CFs and elaborated it. Table 1 presents an overview of the CFs EDWARD uses.

The salience value (SV) of an individual instance (inst) at any given moment is obtained simply by adding the current significance weights of the CFs which have that instance in their scope:

$$\text{SV(inst)} = \sum_{i=1}^{n} \text{significance weight } (\text{CF}_i^{\text{inst}}).$$

Henceforward, we will say that an individual instance is *in context* if its SV is more than 0. The elegance of this particular notion of salience is that it allows for a unified measure of salience, which is determined by an indefinite number of independent factors that can be monitored separately. This architecture differs from the architectures of related work on multimodal interfaces described in the introduction, which all adopt Grosz and Sidner's approach to modeling referents in context. In Section 5, we will compare their approach with ours.

In EDWARD we presently use seven CFs (see Table 1): four serve to model linguistic context effects and three to model perceptual context effects. The linguistic CFs are major-constituent referent CF, subject referent CF, nested-term referent CF, and relation CF. Major-constituent referents are the referents of the subject, the direct object, the indirect object, and the main modifiers of a sentence. They are the role fillers of the relation expressed by the main clause. A major-constituent referent CF has an initial significance weight of 3. (All significance weights have been determined by trial and error and, as will be shown in Section 5, work fine.) Subject referent CFs model the

Table 2
Example of salience value calculation.

	SV of Koen	SV of Ria	SV of the Article
	0	0	0
Koen is de echtgenoot van Ria. Koen is the husband of Ria.	3 + 2 = 5 subject + major	1 nested	0
Hij schrijft een artikel. He writes an article.	(3 − 1 + 2 − 1) + 3 + 2 = 8 (existing) + subject + major	1 − 1 = 0 existing	3 major
Het artikel gaat over zijn vrouw. The article is about his wife.	(3 − 1 − 1 + 2 − 1 − 1 + 3 − 1 + 2 − 1) + 1 = 5 (existing) + nested	3 major	(3 − 1) + 3 + 2 = 7 (existing) + subject + major

observation that referents of subject noun phrases (NPs) are more salient than referents of the other major clause constituents. Their initial significance weight is 2.[3] Nested-term referents are the referents expressed by NP modifiers. These referents are mentioned in the sentence, but they are less prominent than the subject referents or major referents. Nested-term referent CFs have an initial significance weight of 1. Relation CFs are created for all the relations expressed by a sentence, e.g., by the main clause, or by NPs modifying prepositional phrases. Their purpose is to make references to actions expressed in a sentence possible, as in, for example, *do it again*. Their initial significance weight is 3. The decay function of the linguistic CFs subtracts 1 from a CF's weight at each successive update. If a CF's weight equals 0, the CF is discarded.

Table 2 shows how the salience of some individual instances changes in the course of a short dialogue. The three rightmost columns present the SVs after the interpretation of the utterance in the left column. These values are used for the interpretation of the next sentence. After each sentence, the existing CFs are updated by calling their decay functions, and new CFs are created.

The perceptual CFs are as follows: visible referent CF, selected referent CF, and indicated referent CF. Visible referent CFs cause referents that are visible to have a higher SV than referents that are not visible. A visible referent CF has an initial significance weight of 1, so a referent that is visible will be a little more salient than a referent that is not. As soon as the graphical representations (icons) of the referents in the scope of a visible referent CF become invisible (e.g., as a result of a scroll action), the weight drops to 0 and the CF will be discarded. Selected referent CFs cause selected referents to be more salient than referents that are merely visible. A selected referent CF is created when an icon has been selected by the user (by moving the mouse to the icon and clicking the left mouse button), or when the user has requested the system (in natural or formal language) to select icons. Its significance weight is initially 2, and it remains 2 for as long as the icon remains selected. As soon as the icon is deselected, the weight drops to 0 and the CF will be discarded. An indicated referent CF, finally, causes a referent that is indicated by either the system or the user to be very salient for a short time. Indication by the system is done by means of a simulated pointing gesture: a fat, animated growing arrow to a particular icon (for instance, generated upon the question "Which e-mail message is about parsing?"). An indicated referent CF has an initial significance weight of 30 to make sure that the referent in its scope

3 Note that the subject referent resides in two scopes: subject and major-constituent CFs.

Table 3
The four types of referring expressions. Anaphoric expressions are only possible in the NL mode. EDWARD is able to deal with all four types.

	Referring Expressions	
Mode	Deictic	Anaphoric
NL (unimodal)	*this file on the left* *the dissertation*	*it* *this file*
Graphics (unimodal)	*simulated pointing gesture*	–
Multimodal	*this* ↗ *file* *IPF* ↗ *he* ↗ *you* ↗	–

will be the most salient one immediately after the pointing has occurred. After the first update, its significance weight drops to 1, and at the next update, it becomes 0. Notice the difference between selection and indication. Selection is an action only the user can initiate; if the selection is done with a pointing action, both a selected referent CF and an indicated referent CF are created (e.g., for *donald_report* in Figure 2); otherwise only a selected referent CF is created. However, both the user and EDWARD can point, creating indicated referent CFs; pointing has a more temporary effect than selection.

4. Interpreting Deictic and Anaphoric Expressions in EDWARD

EDWARD is able to interpret the two kinds of referring expressions distinguished in the introduction, viz., deictic and anaphoric expressions. When combined with the three categories of interaction modes—unimodal graphical, unimodal linguistic, and multimodal—this results in the four types of referring expressions listed in Table 3.[4]

The basic principle that is used by EDWARD to solve referring expressions is the same for all four types of referring expressions shown in Table 3. Both EDWARD's graphics processes and its syntactic, semantic, and pragmatic interpretation processes operate on line (i.e., interpretation starts directly and goes on while the user enters the remaining of his utterance), incrementally (i.e., the interpretation is built up piece by piece from left to right), and in parallel (i.e., more than one interpretation process can be handled at every moment).

To determine the referent of a phrase, first all individual instances satisfying the semantic restrictions of the phrase are listed. The one with the highest SV, being the most likely referent, is put at the front. Next, after completion of the phrase, the salience of each referent is retrieved by adding the significance weights of all CFs that have this individual instance in their scope.[5] The most salient individual instance is taken to be the referent of the phrase. In the final sentence of Table 2, for example, the referent of the phrase *het artikel* (the article), is the most salient individual instance belonging to the class <article> or to any of its subordinate classes. This approach

4 Referring by name is not included in this table, because it is neither a deictic nor an anaphoric reference. However, EDWARD solves referring by name the same as it does the other four types of referring expressions.

5 The programming language CommonORBIT used in EDWARD provides pointers back from the object to the CFs that have the object in its scope (which compares to Alshawi's notion of marking).

implies that if a particular individual instance has the highest SV, the user need not be very specific and can use, for example, *het* (it), *die* (that one), *die file* (that file), or *dat ding* (that thing). If the highest SV is shared by several instances (a *tie*), EDWARD will ask the user to indicate which of the candidates is intended (e.g., "Do you mean donald_report?"). The following three subsections describe how EDWARD deals with the specifics of the four types of referring expressions in turn.

4.1 Unimodal Linguistic Reference

4.1.1 Anaphora. Anaphoric expressions can be generated using demonstratives: e.g., *dit* (this), *deze files* (these files); personal pronouns, e.g., *hij* (he), *het* (it); and adverbs, e.g., *daar* (there).

To determine the referent of an anaphoric expression, the interpretation component retrieves the most salient, semantically appropriate referent. The salience of a referent is influenced by both linguistic and perceptual context, as was described in Section 3.2.

Plural reference is handled by using sets. To illustrate this, suppose EDWARD has just generated *Het bevat gr2_report en qbgc.* (It contains gr2_report and qbgc.). At that point, a set instance <set#1189> consisting of <spin-report#6362> and <spin-report#6173> is in context, as are the two individual file instances (though they have lower SVs than the set instance). If the user enters *Verwijder die.* (Remove them.), *die* (them) is considered to refer to the most salient instance satisfying the semantic restrictions, in this case <set#1189>.

An interesting subset of anaphoric expressions are *inferential anaphors*. Inferential anaphors are references to individual instances that are not explicitly introduced in the dialogue, but are implicitly introduced by associated instances: e.g., *The secretary* in the sentence pair *The NICI has 80 employees. The secretary is called Hil.* To identify the correct referent, an inference must be made, in this case that institutes employ secretaries. Haviland and Clark (1974) called this type of inference a *bridge*. There are (at least) two ways to have the system "cross the bridge" and resolve inferential anaphors. The first involves the incorporation of associative CFs that create some salience for associates of individual instances just mentioned (e.g., upon mentioning of the NICI, creating associative CFs for the institute's secretary, its director, its hosting university, etc.). We have discarded this option because it is unattractive from a computational point of view. In many domains, the number of associated individual instances of a mentioned individual instance may be very high. Creating associative CFs for all of these associate individual instances is computationally expensive, especially since most of them would have been created without being of any use (only seldom are there several bridges to cross simultaneously). In a worst case scenario, associative CFs interfere with the referent resolution of normal anaphoric expressions. Not-mentioned individual instances that are in the intersection of the sets of associate individual instances of several consecutively mentioned referents may become more salient than instances that have been mentioned. For example, suppose Herb, the brother of the boss of the NICI, and Catherine, the boss's sister, visit the NICI. Upon interpretation of *Herb and Catherine visit the NICI,* the boss of the NICI would have some salience owing to three associate CFs that have been created for it. But any subsequent male pronoun (he, him, his) can refer only to Herb and not to the not-mentioned boss of the NICI.

In the second solution, associate individual instances are not in focus as long as interpretation of referring expressions can work as described above. If no referent can be found by the interpreter for a particular phrase, e.g., no secretary is in context in the case of *The NICI has 80 employees. The secretary is called Hil,* for all referents that are in context, starting with the one with highest salience, their associated individual instances

are retrieved and matched with the class of the phrase. We currently use the following tentative heuristic for associated individual instance retrieval: All relations are taken into account between the referent in context (in this example, <department#276>, having a <*name*> relation with *NICI*) and a referent of the requested class that can be expressed by the lemma *van* (of). In the example, this simulates the NPs: *the secretary of the department*.[6] An advantage of this approach is that referent resolution for phrases other than inferential anaphors is not affected. No effort is wasted in creating associative CFs for individual instances that are not mentioned. Starting the search process at the most salient instance saves computational costs.

4.1.2 Deixis.
Personal deixis. The intension of the personal pronouns *ik* (I) and *jij* (you) is represented using the following predicates:

$$ik \rightarrow \exists_{(x,y)}\, \text{cognizer}(x) \wedge \text{talking-to}(x,y)$$
$$jij \rightarrow \exists_{(x,y)}\, \text{cognizer}(x) \wedge \text{talking-to}(y,x)$$

where the predicate *cognizer* is taken from Pylyshyn (1984), meaning any rational agent, e.g., a person or a dialogue system, and *talking-to* is a predicate that represents the dialogue situation at any time. For example, when the user is entering an input sentence, the clause *talking-to(user, system)* is true so the pronoun *ik* (I) refers to the user. It is the dialogue manager's task to keep track of who is talking to whom and to update the knowledge base accordingly.

Temporal deixis. The interpretation of temporal deixis critically depends on the time of speech of the utterance. EDWARD uses the machine time as an anchoring point. For example, the time interval of the relation <live-in#1> expressed by *Koen woont in Nijmegen.* (Koen lives in Nijmegen.) is an open-ended time interval starting at the machine time at the time of interpretation and ending at *NOW*. If another related relation is added to the knowledge base, e.g., <live-in#2> expressed by a subsequent *Koen woont in Amsterdam.* (Koen lives in Amsterdam.), the open-ended time interval of the first <*live-in*> relation is closed, ending at the current machine time at the time of interpretation of the second relation. The first <*live-in*> relation can now be referred to in simple past tense; the second, in present tense. For example, in case of a subsequent question like *Woont Koen in Amsterdam?* (Does Koen live in Amsterdam?), the time interval of this question relation, viz., *NOW*, is included by the time interval of <live-in#2> found in the knowledge base, and thus the system would respond with *Ja, hij woont er.* (Yes, he lives there.). If, however, the question were *Woont Koen in Nijmegen?* (Does Koen live in Nijmegen?), *NOW* is not included by the time interval of <live-in#1>, the relation no longer holds, and the system would respond negatively. Since the system, in this case, knows what <*live-in*> relation does hold, it can respond cooperatively with *Nee, hij woont in Amsterdam.* (No, he lives in Amsterdam.). Currently, simple present and simple past tense are the only two tenses handled.

Spatial deixis. The presence of a visible model world invites the user to generate referring linguistic expressions involving the spatial environment. We call definite NPs referring to the only object of a certain type visible at that moment implicit spatial

6 The heuristic can be seen as a practical solution to find attributes of a concept and in Dutch seems to work in almost every case. A more general solution consists, of course, of specifying associated concept links in the semantic network, which currently contains only *is-a* links.

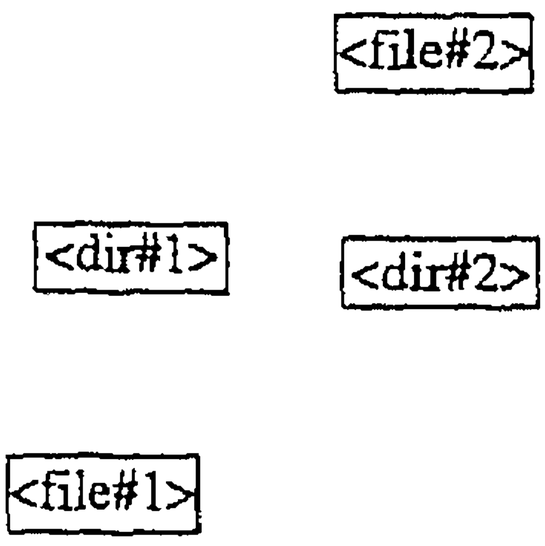

Figure 3
Reference resolution of spatial descriptions: a schematic lay out of two directory icons and two file icons.

deixis. An example is the NP *the closed bookcase* in the case that only one icon resembles a closed bookcase. EDWARD solves this type of referring expression simply by obtaining the most salient object of the right type. The object will be in the scope of the visibility CF, and if no other object of this type is in context, the visible object thus will be selected as the referent.

Explicit references to the spatial environment are references to spatial relations. Spatial relations can be divided into *topological* relations and *projective* relations (Retz-Schmidt 1988). Examples of topological relations are IN, AT, and NEAR. Topological relations (e.g., *the file near it*) refer to topological relations between the referent and the *relatum* (in this example, the object referred to by *it*). Examples of projective relations are IN FRONT OF, BETWEEN, LEFTMOST, and BESIDE. Projective relations convey information about the direction in which an object is located with respect to another object or to the world. A particular linguistic expression describing a projective relation can be used in three different ways: deictically, intrinsically, and extrinsically. The phrase *the ball in front of the car*, for example, can have three interpretations. It could mean that the ball's location is referred to in relation to the car from the speaker's point of view (deictic use), or with respect to the orientation of the car itself (intrinsic use), or with respect to the actual direction of motion of the car (extrinsic use).

In EDWARD all linguistic expressions describing spatial relations are interpreted deictically. For the time being, this restriction does not cause problems. Extrinsic use of, for example, the projective preposition *left of*, i.e., left of an object that is being dragged by the user, when looking in the direction of dragging, is currently impossible since the user cannot drag and write linguistic expressions simultaneously. Intrinsic use of, for example, *left of* and *right of* is assumed to be rare in the current domain: none of the now more than 50 users that have interacted with EDWARD used it.

To determine the referent of a spatial expression, the visible Model World is scanned for a referent, using the intension of the spatial relation and the relatum. The area to be scanned depends on the context. For some relations, the boundaries of the Model World are searched for (e.g., *the bottom most file*); for others, the area in the relatum's vicinity (e.g., *the file left of donald_report*), or the area of the most salient objects (e.g., *the file on the left* if the directory containing that file is very salient) are searched for.

Now let us consider a more complex example. Suppose there are two directory icons and two file icons, positioned as schematically indicated in Figure 3. Suppose all objects have a SV of 1, and no other files and directories are in context (i.e., have a SV greater than 0).

Both expressions, *the file* and *the directory*, are ambiguous and would force EDWARD to start a clarifying user consult. However, the spatial description *the file below the directory* is unambiguous. Relatum and referent support each other in reference solution. EDWARD scans the vicinity of both relata <dir#1> and <dir#2>. Since it finds a referent (<file#1>) only for <dir#1>, it can determine <file#1> as referent and <dir#1> as relatum.

4.2 Unimodal Graphical Reference

Because the notions of deixis and anaphora make sense only in the language mode, we cannot apply this distinction to the action mode. All unimodal graphical reference is considered deictic.

The graphics analyzer interprets the pointing gestures produced by the user. An additional opportunity in simulated pointing that is not available in normal gesturing is the provision of feedback about the success of a pointing gesture. The indicated object, henceforward referred to as the *demonstratum*, (e.g., a file icon, directory tree, or screen position), is marked using reverse video and becomes selected. Usually, the user points to an object to indicate that it is the argument of the command he wants to perform, e.g., a file copy command. Objects remain selected until the user points to another object or explicitly deselects the selected object. The pointing gestures that the system produces have been designed not to interfere with user selection. The graphics analyzer always immediately updates the selected CF of the demonstratum.

The user can simulate *pars-pro-toto* and *totum-pro-parte* pointing gestures. In pars-pro-toto pointing, an object is selected by pointing to a pixel that is within the object's selection area (which encloses the area covered by its icon) and subsequently pressing the select object mouse button. By simultaneously pressing the multiple selection key, multiple objects can be selected. In totum-pro-parte pointing, objects are selected either by enclosing the icons in a mouse-driven rectangle, or by pointing to an icon that is part of a compound object, typically the root of a directory tree, and pressing the select compound object mouse button.

Notice that all simulated pointing gestures are in principle ambiguous: they can refer either to the positions themselves or to the objects located at these positions. When operating in the action mode, i.e., selecting and manipulating graphical representations, the gestures can be taken to refer to the objects at the positions indicated, since screen positions cannot be manipulated.

4.3 Multimodal Referring Expressions

Multimodal deictic referring expressions combine referring linguistic expressions with simulated pointing gestures. Since pointing to time is impossible, only spatial and personal deixis is possible in multimodal referring expression. Demonstrative expressions (e.g., *dit bestand/deze* [this file/this one]) in combination with the realization of an appropriate pointing gesture are common examples of multimodal referring expressions. Notice, however, that demonstrative phrases are not necessarily accompanied by pointing gestures (they can be used anaphorically as well; see Section 4.1.3). Moreover, pointing gestures can also be combined with other, non-demonstrative definite NPs: *Het rapport over DoNaLD zit in Claassen*↗. (The report about DoNaLD is in Claassen↗; with a pointing gesture to the Claassen directory).

To determine the referent(s) of (multimodal) referring expressions, the interpretation component retrieves the most salient referent that satisfies the semantic restrictions of the input phrase. The salience of a referent is influenced by both linguistic and per-

ceptual CFs, so the multimodal referring expressions are solved in exactly the same way as unimodal referring expressions. Consider, for instance, the interpretation of *dit* (this one) in sentence (2a) versus the interpretation in sentence (2b) following the NL command (1):

(1) *Zoek het rapport over* Gr2. (Find the report about Gr2.)

(2a) *Kopieer alle rapporten behalve dit*. (Copy all reports except for this one.)

(2b) *Kopieer alle rapporten behalve dit* ↗. (Copy all reports except for this ↗ one; where the report named donald_report is the demonstratum).

Let us assume that the referent of the report about Gr2 has a SV of 3 just before sentence (2a) or (2b) is interpreted. The referent of *dit* (this one) in sentences (2a) and (2b) would be the most salient report at that moment, which would be the report about Gr2 in sentence (2a), but the report pointed to (donald_report) in sentence (2b). Notice that multimodal expressions with a redundant pointing gesture (e.g., gr2_report ↗ if there is just one object named gr2_report in the context) are solved the same way.

Now, what happens if the user uses multiple pointing gestures within one utterance as in the example *Zet deze file hier* ↗, en *deze* ↗ *daar* ↗. (Put this file here ↗, and this ↗ one there ↗.)? The fact that both EDWARD's graphics processes and its syntactic, semantic, and pragmatic interpretation processes operate on line, incrementally, and in parallel implies that the context effects of a pointing gesture can immediately be taken into account by the reference analysis process. So, if the user points to an icon, the salience of its referent increases immediately, making it the most likely candidate referent of the phrase at hand. By the time the user starts to point a second time, the analysis of the previous multimodal referring expression has been completed, and the context effect of the second pointing gesture is used to solve the corresponding referring expression.

Empirical evidence shows that deictic gestures are indeed exactly coordinated with their associated verbal expressions. Marslen-Wilson et al. (1982), for example, observed that their subject's pointing gestures occurred simultaneously with the demonstrative in the associated NP, or when no demonstrative was used, with the head of the corresponding NP. They report no deictic gestures after completion of the corresponding NP. This implies that the timing of their subject's pointing gestures would satisfy the restriction mentioned above.

Since pointing yields both the screen location pointed to and the object positioned at that location, it is the interpreter's job to disambiguate. Furthermore, more ambiguity arises if two objects have selection areas that partially overlap and the user points in this intersection area. EDWARD cannot determine which object's area the user referred to unless this pointing action is part of a multimodal expression such as *dit* ↗ *boek* (this ↗ book). The graphics interpreter passes all candidates (in this case, for example, <screen-position#798>, <book#248>, <report#546>) on to the dialogue manager, which brings them temporarily in the context. That is, an indicated CF and a selected CF are created for each of them. Guided by the language interpreter, the dialogue manager then decides which of the referents was intended. In the case of *dit boek*, pointing to a report or screen location was not intended, and thus the dialogue manager decides that the indicated CFs and selected CFs update of the report and screen location were invalid. It kills these CFs and subsequently deselects the unintendedly selected object.

5. Assessing the Quality of EDWARD's Referent Resolution Model

To assess the quality of EDWARD's referent resolution model, we collected a series of referring expressions, which were processed by three different referent resolution models, namely that of EDWARD, as described above, a very simplistic model, and the sophisticated and often applied model proposed by Grosz and Sidner (1986). Since there are no benchmarks available to evaluate referent resolution models, we had subjects interact with EDWARD to compile a set of referring expressions. Usually, NL test sentences are made up by evaluators/designers themselves, but we think made-up test sentences may to some extent be unconsciously biased. In the course of developing EDWARD's referent resolution model, we used hundreds of test sentences made up by ourselves to debug and test the program. Real referring expressions, generated by users not familiar with the internal processes of the interpreter, provide a more solid empirical basis for evaluation. In Section 5.1, we present an overview of these user-generated referring expressions. In Section 5.2, we briefly describe the way the two alternative referent resolution models work. The results of feeding the test sentences to the three different referent resolution models are given in Section 5.3. In assessing the quality of a referent resolution model, it is, however, also necessary to analyze the internal affairs of the model and determine the inherent limitations that follow from its design. In Section 5.4, we present the inherent limitations of EDWARD's referent resolution model as well as those of the two alternative models.

5.1 A Test Set of Referring Expressions

By having five subjects (two men and three women) interact with EDWARD, we obtained a total of 125 real, user-generated referring expressions. The subjects all had some previous experience with the system, but this was limited to 1 or 2 hours and dated from 2 to 3 months before. None of them had knowledge of the internal affairs of EDWARD's referent resolution model. The subjects were to perform 19 tasks; most were information retrieval tasks, but some tasks involved effectuating a change in the file system. The subjects were not informed which words and syntactic and semantic constructs could be handled by the system and which could not, but they all knew from their previous encounters that the system was not an unrestricted NL interface. We did explicitly encourage the subjects to use the shortest referring expression possible whenever they felt it was appropriate. From earlier experiments with EDWARD (Huls and Bos 1993; Huls et al. 1993), we know that some users are reluctant to use referring expressions other than by name (probably due to the impact of command language interfaces for familiar file management systems). Examples of the tasks the subjects were to perform are the following:

1. Find out who is the boss of the NICI; followed by

   Find out who is the secretary of the NICI.

2. Find out who live in Nijmegen; followed by

   Find out whether all women living in Nijmegen work at the NICI.

3. Put a copy of this [*experimentor points at leftmost file on screen*] file in this [*experimentor points again*] directory.

These tasks were supposed to induce inferential anaphors (1), plural referring expressions (2), spatial deixis (3), and multimodal referring expressions (3).

As we expected, different subjects performed the tasks differently. Some, for example, needed two questions to find out who is the secretary of the NICI, others just one, of which two subjects indeed used the induced inferential anaphor. Table 4 shows several translated sample sentences taken from the set of sentences the five subjects keyed in to perform the 19 tasks. To show the variety in use of referring expressions, we present under (a) the sentences with the largest amount of deictic and anaphoric expressions keyed in by the subjects and under (b) the least amount. For example, (19a) shows the sentence subject #4 used for task 19, with two pronouns, and (19b) shows subject #3's sentences with only one pronoun:

The frequency with which the different types of referring expressions occurred can be found in Table 5. Here a clearer view on the variety among subjects in the way of referring is presented. (The types of referring expressions of Table 5 do not exactly match the four types mentioned in Table 3. Unimodal graphical deixis was not encouraged in the experiment and therefore did not occur; reference by name occurred frequently, but this type of reference is not considered to be deictic or anaphoric, and their interpretation is therefore less interesting from a computational linguistics point of view.)

Finally, we present some data on the frequencies of use of the two most common words that can feature in both deictic and anaphoric expressions, viz., *dit* and *deze* (two demonstrative pronouns, respectively neuter and non-neuter). Table 6 shows the variety in use.

5.2 Two Alternative Referent Resolution Models

The sentences with the referring expressions as described in the previous section were processed by EDWARD's referent resolution model and two alternative referent resolution models. The first alternative model is a very simplistic one. It simply takes the last mentioned semantically appropriate referent. For example, in the sequence *The secretary is Hil. Where does she live?* the pronoun *she* is taken to refer to the last mentioned female, in this case Hil. We implemented this Simplistic Model and provided EDWARD with a switch to determine whether sentences should be processed either with the original Context Model or with this alternative Simplistic Model. Each referent mentioned in the dialogue is put on a stack, and when interpreting a referring expression, the stack is processed from top to bottom. To prevent uncontrolled growing of the stack, we had the system discard the object at the bottom of the stack as soon as the stack length exceeded a certain maximum.

The second alternative referent resolution model is that of Grosz and Sidner (1986). Their model consists of two separate mechanisms, each resolving a specific type of referring expression. The first mechanism is called *focusing*. Focusing is used to limit the information that must be considered in identifying the referents of certain classes of definite NPs. A stack is created in which the focus spaces corresponding to the discourse segment purposes are stored. All entities mentioned in a discourse segment purpose and all related entities (e.g., parts of mentioned entities) are stored in a focus space. New focus spaces are put on top of the focus stack, and the referent for a NP will be searched from the top down. For our data analysis, we suppose a new discourse segment purpose for each new sentence. A second mechanism, called *centering* (or *immediate focusing*), is used for pronoun resolution. In brief, a backward-looking center is associated with each utterance in a discourse segment. Of all focused elements, the backward-looking center is the one that is central in that utterance. A combination of syntactic, semantic, and discourse information is used to identify the backward-looking center. The fact that some entity is the backward-looking center is used to constrain the search for the referent of a pronoun in a subsequent utterance. Unfortunately,

Table 4
A translated compilation of the sentences the subjects used to perform the tasks. Under (a), the sentences with the largest amount of deictic and anaphoric expressions are given; under (b), the sentences with the least amount of deictic and anaphoric expressions are given.

User's Input	EDWARD's Output
1a Who is the boss of the NICI?	The boss of the NICI is Charles.
2a Who is the secretary? b Who is the secretary of the NICI?	The secretary is Hil.
3a Where does she live? b Where does the secretary of the NICI live?	She lives in Nijmegen.
4a Who else lives there?[a] b Who else lives in Nijmegen?	Charles, Carla, Gerben, Alice, Gerard, Elena, Francois, Edwin, Sacha and Ger live there.
5a Do the women work for the NICI? b Do the women who live in Nijmegen work for the NICI?	Yes, they do.
6a Does Wim live in Nijmegen?	No, he lives in Lent.
7a What is the name of his wife?	The name of his wife is Hanneke.
8a What did he write? b What did Wim write?	He wrote donald_report and conceptuele_generatie.
9a Copy these files. b Copy donald_report and conceptuele_generatie.	Ok.
10a What is the subject of this↗file?	The subject of this file is DoNaLD.
11a Move it to Bos. b Move donald_report to Bos.	Ok.
12a Close this. b Close Bos.	Ok.
13a Who sent the left e-mail? b Who sent this↗e-mail?	Wietske sent it to Carla.
14a Which e-mails are sent by Alice?	Alice mailed this↗e-mail to Wim, this↗e-mail to Koen and this↗e-mail to Carla.
15a She sent an e-mail about Bos to Wietske. b Alice sent an e-mail about Bos to Wietske.	Ok.
16a Remove this↗. b Remove this↗e-mail.	Ok.
17a Move all e-mails of her to Dijkstra. b Move all e-mails of Alice to Dijkstra.	Ok.
18a Her husband is called Lou. b Alice's husband is called Lou.	Information added.
19a He lives in her hometown. b He lives in the same town as Alice.	Information added.

a The original Dutch in 4a and 4b uses the third-person plural form.

Table 5
Numbers of occurrences of different types of referring expressions.

Type of Referring Expression	Subject #1	#2	#3	#4	#5	Total
By name	18	14	13	12	16	73
Normal anaphor	4	8	10	12	8	42
Inferential anaphor	0	1	1	0	0	2
Unimodal NL deictic	0	0	0	1	0	1
Multimodal deictic	2	2	2	0	1	7
Total	24	25	26	25	25	125

Table 6
Numbers of occurrences of the use of *dit* and *deze* (both meaning *this*) anaphorically and deictically.

Type of Referring Expression	Subject #1	#2	#3	#4	#5	Total
dit + *deze* anaphorically	0	1	2	5	2	10
dit + *deze* deictically	2	2	2	0	1	7
Total	2	3	4	5	3	17

Table 7
The number of referring expressions interpreted correctly by the three referent resolution models.

Referent Resolution Models	Subject #1	#2	#3	#4	#5	Total
EDWARD's Context Model	24	25	26	25	25	125
Simplistic Model	23	24	25	23	24	119
Grosz and Sidner	24	25	26	25	24	124

Grosz and Sidner's model presupposes several sorts of information at moments when EDWARD's interpreter does not have these available. Consequently, we could use only a pen and paper analysis of how their model processes the test set of referring expressions.

5.3 How the Referent Resolution Models Dealt with the Test Set

The sentences with 125 referring expressions entered by the five users to perform the 19 tasks were processed by the three referent resolution models. Table 7 shows the scores.

5.3.1 Context Model. EDWARD's Context Model determined the right referent in all 125 referring expressions. However, in the session of subject #2, we discovered an

error in the interpretation of a dozen sentences this subject keyed in just for curiosity *after she had completed the 19 tasks*. She continued as follows:

Alice wrote him an e-mail.	OK, information added.
Put that e-mail in Dijkstra.	OK, information added.
Is this↗the e-mail to Lou?	No, this↗one is.
Which one?	This↗one.
What is in this e-mail?	Sorry, please rephrase.
What is the topic of this e-mail?	I don't know.
Where is *her* e-mail to Lou?	

Here, the referent of the pronoun *her* was mentioned too long ago for EDWARD to be able to locate the referent Alice. EDWARD therefore had to ask the user *Whom do you mean with 'her'?*

5.3.2 Simplistic Model. The results of the simplistic referent resolution model were surprising: we counted only 6 misses. Task (15) particularly showed the restrictions of the Simplistic Model:

(14) Which are the e-mails sent by Alice?

Alice mailed this↗e-mail to Wim, this↗e-mail to Koen and this↗e-mail to Carla.

(15) *She* sent an e-mail about Bos to Wietske.

The *she* of (15) is considered to refer to Carla, the last mentioned female, but the user actually referred to Alice. Similar problems occurred with *this* in task (16).

5.3.3 Grosz and Sidner. Using a pen and paper analysis of how the Grosz and Sidner Model processes the sentences, we think their model resolves all but 1 referring expression correctly. The only problem we encountered concerned the use of two pronouns in one sentence: *he* lives in *her* town. The original model excluded these double occurrences.

5.4 Inherent Limitations of the Referent Resolution Models

In this section, we describe several problems of the three reference resolution models that follow from their design but did not become apparent in the test set evaluation.

First, EDWARD's Context Model and the Simplistic Model do not make any predictions about discourse intention. Discourse intentions play a primary role in explaining discourse structure, defining discourse coherence, and providing a coherent conceptualization of the term "discourse" (Grosz and Sidner 1986). Discourse intentions can provide clues for the beginning and ending of dialogues and subdialogues. Referent resolution can make use of this structure to exclude referents to (sub)dialogues that are ended. Furthermore, subdialogues do not interfere with the referent resolution of the main dialogue. Grosz and Sidner's theory of discourse structure, on the other hand, does address these problems.

The Context Model obviously still lacks several factors that can influence the salience of a referent. An example is the different context effects of reference by a pronoun versus reference by a definite full-fledged NP. Grosz and Sidner mention this distinction but do not, however, provide a thorough analysis of all syntactic, semantic, and pragmatic rules they envisage to play a role in either focusing or centering.

A problem for all three of the referent resolution models is the resolution of cataphors. In contrast with anaphors, cataphors refer to instances that will be introduced later in the discourse (e.g., *He will win who...*). All three models will (try to) locate the referent of *he* in the set of individual instances mentioned before. The resolution of cataphors, however, requires a more lazy evaluation.

6. Conclusions

We have collected some indications about the quality of the Context Model for referent resolution we implemented in our multimodal user interface EDWARD. We have compared the capabilities of this model with two alternative models, both empirically, using a test set of 125 user-generated referring expressions obtained from interactions with EDWARD and, analytically, studying the inherent limitations that follow from the models' designs.

On empirical grounds we conclude that the Simplistic Model, in which anaphoric expressions are considered to refer to the last mentioned semantically appropriate object, is inadequate. Though it performed, by far, better than we anticipated, too often the wrong object is taken to be the referent.

The quality of the other alternative model for referent resolution, the Grosz and Sidner Model, seems to compare to the quality of EDWARD's Context Model. As we understand the Grosz and Sidner Model, it processed 124 referring expressions correctly (but this may be inaccurate, since we do not have an implemented version of the model at our disposal). Furthermore, it will have problems with interpreting cataphora properly. EDWARD's Context Model performed well on all 125 test expressions, but cataphora will also be misinterpreted. The Grosz and Sidner Model has a much broader scope. In particular, their model addresses the notion of discourse coherence. It would be interesting to explore how the insights of Grosz and Sidner with respect to discourse coherence can be used to elaborate EDWARD's Context Model to render it able to deal with subdialogues.

EDWARD's Context Model differs significantly from the Grosz and Sidner Model from an engineering and computational point of view. The Context Model is relatively simplistic. EDWARD never needs to figure out the type of an expression that is being analyzed: for all referring expressions, the most salient referent is chosen. Moreover, entities and relations are handled in a uniform fashion, and syntactic as well as perceptual influences on salience are incorporated into one model. The general applicability of the technique adds to its beauty. The language generation component uses it as well. Both components use the role-filler class restrictions, the cardinality information, and the role-set restrictions from the knowledge base, and they use the same CFs, with the same initial significance weights, and the same decay functions of the Context Model. Grosz and Sidner propose a complex system of rules. In the Context Model, on the other hand, influences originating from different levels and types of processing are modeled by individual CFs, which are created and managed locally, i.e., by these processes themselves. As a result, the influences on an object's salience are represented distributed and independently, which is attractive from a computational point of view. Furthermore, the addition of new CFs, which would require explicit detailed changes in Grosz and Sidner's rules, will be easier because the procedures that use the salience information can stay exactly the same.

Though our empirical and analytical studies were only small and provide no firm basis for drawing conclusions, we do find indications that the quality of EDWARD's Context Model compares to a large extent to the quality of the more complex Grosz and Sidner Model. Therefore, if one is in need of a referent resolution model for a particular

NL interpreter in a setting where subdialogues are rare, we think that EDWARD's Context Model is a good alternative to the complex rule system of Grosz and Sidner. The model is easy to build, to maintain, and to expand, and it is computationally fairly inexpensive.

Acknowledgments

This research was performed within the framework of the research programme 'Human-Computer Communication using natural language' (MMC). The MMC programme is sponsored by SPIN Stimuleringsprojectteam Informaticaonderzoek, Digital Equipment B.V., Sun Microsystems B.V., and AND Software. We wish to thank Koenraad De Smedt, Gerard Kempen, and three anonymous reviewers of *Computational Linguistics* for their helpful comments on a preliminary version of this paper.

References

Allgayer, Jürgen; Jansen-Winkeln, Roman; Reddig, Carola; and Reithinger, Norbert (1989). "Bidirectional use of knowledge in the multi-modal NL access system XTRA." In *Proceedings, 11th International Joint Conference on Artificial Intelligence*, Detroit, Michigan. 1492–1497. Los Altos, California: Morgan Kaufmann.

Alshawi, Hiyan (1987). *Memory and Context for Language Interpretation*. Cambridge: Cambridge University Press.

Bos, Edwin (in press). "A multimodal graph-editor." In *Syntax-Directed Editing*, edited by L. Neal and G. Szwillus. New York: Academic Press.

Bos, Edwin; Huls, Carla; and Claassen, Wim (1994). "EDWARD: Full integration of language and action in a multimodal user interface." *International Journal of Human-Computer Studies* 40:473–495.

Brachman, Ronald, and Schmolze, James (1985). "An overview of the KL-ONE knowledge representation system." *Cognitive Science* 9:171–216.

Claassen, Wim (1992). "Generating referring expressions in a multimodal environment." In *Aspects of Automated Natural Language Generation*, edited by R. Dale, E. Hovy, D. Rösner, and O. Stock, 247–262. Berlin: Springer.

Claassen, Wim, and Huls, Carla (1991). "DoNaLD: A Dutch natural language dialogue system." SPIN/MMC Research Report no. 11, NICI, Nijmegen, The Netherlands.

Claassen, Wim; Bos, Edwin; and Huls, Carla (1990). "The Pooh way in human-computer interaction: Towards multimodal interfaces." SPIN/MMC Research Report no. 5, NICI, Nijmegen, The Netherlands.

Claassen, Wim; Bos, Edwin; Huls, Carla; and De Smedt, Koenraad (1993). "Commenting on action: Continuous linguistic feedback generation." In *Proceedings, International Workshop on Intelligent User Interfaces*, Orlando, Florida. 141–148.

Cohen, Philip (1992). "The role of natural language in a multimodal interface." In *Proceedings, Fifth Annual ACM Symposium on User Interface Software and Technology*, Monterey, California. New York: ACM Press.

De Smedt, Koenraad (1987). "Object-oriented programming in flavors and CommonORBIT." In *Artificial Intelligence Programming Environments*, edited by R. Hawley, 157–176. Chichester: Ellis Horwood.

Grosz, Barbara J. (1978). "Discourse knowledge." In *Understanding Spoken Language*, edited by D. Walker, 229–345. New York: North-Holland.

Grosz, Barbara J., and Sidner, Candace L. (1986). "Attention, intentions, and the structure of discourse." *Computational Linguistics* 12:175–204.

Hajičová, E. (1987). "Focusing: A meeting point of linguistics and artificial intelligence." In *Artificial Intelligence. II: Methodology, Systems, Applications*, edited by P. Jorand and V. Sgurev. Amsterdam: Elsevier Science Publishers.

Haviland, Susan E., and Clark, Herbert H. (1974). "What's new? Acquiring new information as a process in comprehension." *Journal of Verbal Learning and Verbal Behavior* 13:515–521.

Huls, Carla, and Bos, Edwin (1993). "EDWARD: A multimodal interface." In *Proceedings*, TWLT5 Enschede, The Netherlands. 89–98.

Huls, Carla; Bos, Edwin; and Damen, Han (1993). "Fully integrated multimodality: A case study." Paper presented at HCL International '93, August 8–13, Orlando, Florida.

Lyons, John (1977). *Semantics*. Volume 2. London: Cambridge University Press.

Marslen-Wilson, William; Levy, Elena; and Tyler, Lorraine K. (1982). "Producing

interpretable discourse: The establishment and maintenance of reference." In *Speech, Place, and Action*, edited by R. J. Jarvella and W. Klein. Chicester: John Wiley and Sons Ltd.

Neal, Jeanette G., and Shapiro, Stuart C. (1991). "Intelligent multi-media interface technology." In *Intelligent User Interfaces*, edited by J. W. Sullivan and S. W. Tyler, 11–43. New York: ACM Press.

Pylyshyn, Zenon W. (1984). *Computation and Cognition*. Cambridge, Massachusetts: MIT Press.

Retz-Schmidt, Gudula (1988). "Various views on spatial prepositions." Bericht Nr. 3. Universität des Saarlandes, SFB 314 (VITRA).

Sijtsma, Wietske, and Zweekhorst, Olga (1993). "Comparison and review of commercial natural language interfaces." In *Proceedings, TWLT5*. Enschede, The Netherlands. 43–58.

Stock, Oliviero (1991). "Natural language and exploration of an information space: The ALFresco interactive system." In *Proceedings, 12th International Joint Conference on Artificial Intelligence*, Sydney, Australia, 972–978. Los Altos, California: Morgan Kaufmann.